Our Ref.: **1561-71**

U.S. PATENT APPLICATION

Inventor(s): Yves BOUDREAULT
Daniel J. KRAUS

Invention: MIXED-MEDIA DATA ENCODING

NIXON & VANDERHYE P.C.
ATTORNEYS AT LAW
1100 NORTH GLEBE ROAD
8TH FLOOR
ARLINGTON, VIRGINIA 22201-4714
(703) 816-4000
Facsimile (703) 816-4100

SPECIFICATION

Mixed-Media Data Encoding

Background of the Invention

1. Field of the Invention

[0001] The present invention relates to mixed-media data encoding, mixed-media data decoding and a format for the transfer of mixed-media data in encoded form.

2. Description of the Related Art

[0002] As used herein, mixed-media data is defined as the data that is read by a computer program in order to generate a media output in the form of images and sounds. A mixed-media data may include many data types, such as motion capture data, model data, deformation parameters, constraints, expressions or relations, textures, colour values, cameras, lights, video, audio, device information, a timeline or other data types in any combination. Thus, the data is not image data as such but is data that is used and processed and in order to obtain media output.

[0003] A problem with the dissemination of mixed-media data of this type is that the data itself represents highly valuable product and as such it would be highly undesirable for the data to be intercepted by unauthorised parties. It is also possible for a situation to arise in which an external party is allowed to do some things with the data while not being allowed to do other things with the data. In particular, organisations may also be very sensitive to the risks of modifiable three dimensional data being made available to none authorised users.

[0004] Encryption techniques are known for encrypting a file so as to prevent unauthorised access to that file. After a file has been encrypted, it is not possible for anyone to do anything with the file. Similarly, once the file has been decrypted it is then possible for anyone, with appropriate technology, to do anything with the file. Thus, a problem exists in that a technical mechanism is required to control the degree to which external parties may access and manipulate mixed-media data.

[0005] It is known to transfer machine readable data files in a protected form using cryptography techniques, however, a problem with known techniques is that they consider a file to be either encrypted or not encrypted. When encrypted, it is not possible for anyone to do anything with the file and when decrypted it is possible for anyone to do anything with the file. However, media data files include many components and many processes may be performed upon these components. Furthermore, some of these components are more sensitive than others.

Brief Summary of the Invention

[0006] According to an aspect of the present invention, there is provided mixed-media data encoding apparatus, in which said mixed-media data includes a plurality of data types. The apparatus comprises encoding means configured to encode the mixed-media data to prevent unauthorised access and storage means configured to store the encoded data. User access to the data is possible in response to an accessing activity performed by a user. A first set of user modifications may be made to said data (a first level of

access) in response to a first accessing activity and a second set of user modifications may be made to said data (a second level of access) in response to a second accessing activity.

Brief Description of the Several Views of the Drawings

[0007] *Figure 1* shows an environment for the creation, distribution and delivery of mixed-media data;

[0008] *Figure 2* shows a computer animation system;

[0009] *Figure 3* details the computer system identified in *Figure 2*;

[0010] *Figure 4* shows procedures for an encoding operation performed by the system shown in *Figure 2*;

[0011] *Figure 5* shows procedures performed for an encoding operation;

[0012] *Figure 6* illustrates a frame of an animation sequence;

[0013] *Figure 7* shows details the encoding process shown in Figure 5;

[0014] *Figure 8* details the process for combining data elements identified in *Figure 7*;

[0015] *Figure 9* illustrates a detailed example of the three-dimensional data file;

[0016] *Figure 10* shows a table for the definition of passwords;

[0017] *Figure 11* illustrates the procedure for adding passwords identified in *Figure 7*;

[0018] *Figure 12* illustrates the process for combining keys identified in *Figure 7*;

[0019] *Figure 13* illustrates the data encryption process identified in *Figure 7*;

[0020] *Figure 14* illustrates the header creation and addition processes identified in *Figure 7*;

[0021] *Figure 15* illustrates distribution modes for a coded export file;

[0022] *Figure 16* illustrates a decoding operation;

[0023] *Figure 17* illustrates a display requesting a password to be entered;

[0024] *Figure 18* details the process for decoding an encoded file identified in *Figure 16*;

[0025] *Figure 19* illustrates the overall process for receiving an encoded file; and

[0026] *Figure 20* illustrates a frame of an animation generated from the received animation data.

Best Mode for Carrying Out the Invention

Figure 1

[0027] An environment for the creation, distribution and consumption of media data is illustrated in *Figure 1*. In the environment shown in *Figure 1*, most data transfers take place over the World Wide Web **101**, although mechanisms for data transfer, including local networks and physical storage media are equally valid.

[0028] A major producer of cinematographic works, including computer animation, has an animation studio **102** in California, an effects studio **103** in London and a texture studio **104** in Tokyo. In addition, a strong customer base exists in New York therefore a facility **105** is also included in New York for making presentations to potential customers etc.

[0029] During a typical production process, project work is initiated at animation studio **102** whereafter a data file is transmitted to texture studio **104** for two-dimensional textures to be added. In parallel with this or thereafter, data files are transmitted to effects studio **103** for special effects and video components to be included.

[0030] At any stage of the creative process, a file representing work in progress may be conveyed to the customer centre **105**. However, it is appreciated that the customer centre **105** is not particularly secure, compared to studios **102**, **103** and **104**. Furthermore, data files may be given to potential customers to allow them to view them on their own premises. Consequently, although it should be possible to render and view images at the customer centre **105**, it is undesirable for the data supplied to the customer centre **105** to be modifiable, particularly at the three-dimensional level. However, it is necessary in order for the work to be completed for the data files to be modifiable at the three studios. In particular, animation studio **102** must be given full access to the data whereas studio **103** must at least be able to modify video components and studio **104** must at least be able to modify texture components.

[0031] Finished works of animation are distributed to consumers, such as consumers **111**, **112** and **113**. In addition to receiving product by the World Wide Web, as illustrated in *Figure 1*, consumers may also receive product as part of digital television broadcasts via a distributor **115**.

[0032] From the perspective of the material author or distributor, entity **117** is considered to be a consumer. However, entity **117** do themselves provide modified product to consumers **121**, **122** and **123** etc, which are then consumers to the distributor **115** once removed. Consequently, entity **117** is considered within this environment as a "prosumer" ie professional-consumer. The prosumer **117** pays royalties to the distributor that are higher than the royalties paid by consumers **111** etc. The prosumer is provided with a greater level of access to the three-dimensional data, although this access is not as great as that available to the animation studios.

[0033] For the purposes of this disclosure, it is assumed that animation studio **102** generates an animation that is partially complete. The animation data is then encoded and transferred to texture studio **104** in Tokyo. In Tokyo, the data is decoded, manipulated further, encoded again and then returned to animation studio **102**. Similarly, at studio **102**, the received data is decoded, manipulated further and then placed in a condition ready for distribution. Thus, animation studio **102**, in this example, will be performing both an encoding process, to allow secure distribution of data to colleagues and will then also perform a decoding process in order for further manipulations on input data received from colleagues. The encoding process and decoding process will therefore be described with respect to a hardware facility provided at the animation studio **102** although it should be appreciated that similar equipment would exist where other users are being allowed access to the data.

Figure 2

[0034] A computer animation system based at the animation studio **102** is shown in *Figure 2*. The animation system shown in *Figure 2* is configured to manipulate three-dimensional data having many elements, to combine these elements into a mixed-media file format and to facilitate the distribution of three-dimensional data files in an encoded form.

[0035] The system shown in *Figure 2* includes a programmable computer **201** having a DVD drive **202** for receiving CD ROMS and DVDs **203**, along with a disk drive **204** for receiving magnetic disks **205** such as zip disks. Computer system **201** may receive program instructions via an appropriate CD ROM **203** and data files may be transferred using zip drive **205** or via a network connection **206** connected to the World Wide Web **101**. Output data is displayed on a visual display unit **207** and manual input is received via keyboard **208** and a mouse **209**.

Figure 3

[0036] Computer system **202** is illustrated in *Figure 3*. The system includes a Pentium III central processing unit **301** operating under instructions received from random access memory **303** via a system bus **302**. Memory **303** comprises one hundred and twenty megabytes of randomly accessible memory and executable programs which, along with data, are received via bus **302** from a hard disk drive **309**. A graphics card **304**, an input/output interface **305**, a network card **306**, a zip drive **307** and a DVD drive **308** are also connected to bus **302**. Graphics card **304**

supplies graphical data to visual display unit **207** and the input/output interface **305** receives input commands from a keyboard **208** and from a mouse **209**. Zip drive **307** is primarily provided for the transfer of mixed-media data and the DVD drive is primarily used for the loading and installation of new executable instructions, usually provided on CD ROM.

[0037] The preferred embodiment of the present invention provides for the creation and manipulation of three-dimensional image data followed by an encoding process to allow the data to be transferred to other stations in a protected form. For the purposes of this disclosure, this first stage will be referred to as an encoding operation. Similarly, it is also possible for the station shown in *Figure 2* to receive encoded data, decode this data and then perform further manipulations upon the data. For the purposes of this disclosure, such a process will be referred to as a decoding operation.

Figure 4

[0038] The philosophy of the operating environment is illustrated in *Figure 4*. The system includes a plurality of individual programs **401**, **402**, **403**, **404**, **405** etc that facilitate the manipulation of various types of data. Thus, for example, program **401** may relate to the creation and manipulation of animation data, program **402** may relate to the creation and manipulation of model data and program **403** may relate to the creation and manipulation of texture data. Once created in its original format, three-dimensional data is constrained within the environment as illustrated by boundary **411**. It is possible for manipulated data to leave boundary **411** but it achieves this by being processed through an encoding operation **412**.

Thus, any data that does leave the system, by being written to a file or transmitted over a network, does so in encoded form after being processed by the encoding operation **412**.

[0039] Similarly, encoded data may be received and is then manipulated by being processed through a decoding operation **413**. The decoding operation **413** understands how to perform relevant decrypting processes that have been applied to data by an encoding process. The extent to which a user may then perform manipulations upon the decoded data is determined by the user's ability to identify access codes which themselves have been embedded within the encoded data.

[0040] Thus, data may be manipulated and then exported as part of an encoding operation. Similarly, coded data may be received, decoded and then manipulated as part of the decoding operation.

Figure 5

[0041] Procedures performed by the system shown in *Figure 2* for an encoding operation are shown in *Figure 5*. At step **501** the encoding operation program is opened and at step **502** mixed-media data is created. Procedures for creating and manipulating mixed-media data are well known and the exact nature of the processes performed at step **502** are not relevant to the particular nature of the present invention. However, it should be understood that the processes performed at step **502** result in the generation of three-dimensional data and other data types, that requires a further rendering process in order to produce viewable two-dimensional

images.

[0042] At step **503** a question is asked as to whether a file is to be exported. The encoding operation performed by the system shown in *Figure 2* maintains local information in a secure form and does not allow general file copying. In order for a file to leave the system, either over a network connection or via recordable media, it is necessary for the file to be exported and as such the file undergoes an encoding process in order for the export to be completed. Consequently, if the question asked at step **503** is answered in the affirmative, the data is encoded at step **404** and then a file export is performed at step **505**.

[0043] Thereafter, at step **506** a question is asked as to whether new data is to be considered. If the question asked at step **503** is answered in the negative, control is directed to step **506**. If the question asked at step **506** is answered in the affirmative, control is returned to step **502** allowing further data to be created. If the question asked at step **506** is answered in the negative, the program is closed at step **507**.

Figure 6

[0044] Process **502** results in the creation of mixed-media data including three-dimensional data and a resulting three-dimensional animation may be viewed and on monitor **207**, as shown in *Figure 6*. In order for the three-dimensional data to be viewed, individual video frames are rendered and then displayed on the monitor **207**. A model **601** appears to move on a frame-by-frame basis in response to data defining the

animation to be performed. Consequently, the animation comprises data which may have been created by effects studio **103**. Movement of the model 601 is defined by animation data and the animation data may have been created by animation studio **102**. The model **601** appears solid with a outer surface or texture and separate texture data may have been generated by texture studio **104**. Step **502** for the creation of three-dimensional data therefore involves the manipulation of the animation data, the model data and the texture data along with other modifiable parameters or user data to achieve the overall effect. Thus, elements may be defined by a user which may or may not be three dimensional, including texture or material values on polygon vertices.

Figure 7

[0045] The encoding process **504** is shown in *Figure 7*. At step **701** individual data elements, representing different types of data, are combined to form a single file which may then be encoded to produce an encoded file format that is recognisable by decoding processes.

[0046] Once received, the level of access made available to various components of the file is controlled by access codes; these being codes that are usually established by the original creator of the file. Thus, at step **702** the process receives access passwords and at step **703** the access passwords are added to the data.

[0047] At step **704** the process receives a file encryption key and at step **705** the file encryption key is combined with a program key; the latter

being a key that is known to the program and is unique for each version of the program. These two keys are combined and then an encryption process is performed at step **706** on the basis of the combined key produced at step **705**.

[0048] At step **707** the file key is added to a file header and then the header is added to the file itself at step **708** to produce a file that may be exported under process **505**.

Figure 8

[0049] Process **701** for the combining of data elements is illustrated in *Figure 8*. In this example, animation data **801** has been produced and manipulated under the control of program **401**. Similarly, model data **802** has been produced and manipulated under the control of program instructions **402** and texture data **803** has been produced and manipulated under the control of program **403**. The data sets **801**, **802** and **803** are relevant to the particular type of data being produced. A file format combiner process **804** provides, in combination with the system hardware, a means for combining these data formats into a combined binary file identified as a mixed-media file **805**.

Figure 9

[0050] A detailed example of a mixed-media file is illustrated in *Figure 9*. At **901**, the file includes binary data relating to the animation. An animation is created by combining animation data with model data such that a specified model effects movement in accordance with the specified

animation.

[0051] Model data is stored at **902** and this may be defined in terms of polygons or splines.

[0052] At **903** deformation parameters are stored that define the weight relationship between three dimensional elements, such as between a skin and an inverse kinematics bone of an animation model.

[0053] At **904** constraints data is stored defining the relationship between objects or elements.

[0054] At **905** expressions and relations are stored in the form of mathematical relationships between objects and elements.

[0055] At **906** textures, in the form of two-dimensional images or three-dimensional geometric deformations based upon procedural algorithms are stored.

[0056] At **907** colour values are stored with definitions of cameras being stored at 908 and definitions of lights being stored at **910**.

[0057] Location **911** provides for the storage of video clips, taking the form of two-dimensional bit streams. Similarly, at **912** there is the provision for the storage of audio bit streams. Device information such as parameter controls for input and output devices are stored at **913** and timeline

information is stored at 914. In addition, user defined data is stored at **915**.

Figure 10

[0058] After the individual file elements have been combined by process **804** to produce a mixed-media data file **805** access passwords are received at step **702**. The passwords are received in response to a table being displayed on monitor **207** as illustrated in *Figure 10*. The table shown in *Figure 10* is presented to an originator in order to specify the passwords for particular levels of access. In this respect, the program may be provided in two forms. In a first form, as disclosed with respect to the present embodiment, the levels of access are predefined.

[0059] Thus, in the first embodiment, a complete system is provided in which levels of access are pre-specified. However, in an alternative environment, the essential components of a system could be licensed to a commercial vendor, thereby allowing said vendor to define their own levels of access that are consistent with their own distribution procedures and format.

[0060] In the present example, seven levels of access are provided as shown in column **1001**. Column **1002** identifies the type of access associated with each level and, in the enhanced embodiment, the specific type of access provided at each level may be specified by the originator. Column **1003** then provides for a password to be identified for each particular level.

[0061] In this example: Level A provides for the rendering and playback of three-dimensional data so as to produce two-dimensional output. Usually, there is no problem in terms of allowing this level of access therefore it would be usual to leave the password entry blank.

[0062] Level B access allows behaviour triggering to be modified and is secured by password "WORD B". Level C access allows scene control and is protected by password "WORD C". Similarly, level D access allows clip libraries to be modified, protected by password "WORD D" and Level E allows animation to be edited, accessed by password "WORD E". Similarly, Level F allows model editing and texture editing to be performed, protected by password "WORD F" and Level G allows full control to the data, protected by password "WORD G". The passwords are selected by the originator and would tend to change on a file-by-file basis. As is well recognised in the art, it is appreciated that a greater level of security is obtained by allowing larger passwords to be entered.

[0063] Knowledge of a password for a certain level provides automatic access to all of the lower levels. Consequently, if the user is aware of password "WORD F" it is not necessary to know passwords "WORD B" and the others between these extremes.

[0064] In addition to crescive level access, in which each progressively secure level provides an augmented level of access with access to the preceding levels also being available, the access may be discrete, such

that individual levels may be accessed without access to the other levels. Such a discrete approach is applicable in situations where, for example, modifications are to be made to a texture in a particular studio where only access to the texture controls is only required. Thus, using this discrete approach, it would be possible to provide access to these control while not permitting access to other, often simpler elements of the scene.

Figure 11

[0065] After the passwords have been specified, under control of process **702**, the passwords are added to the three-dimensional data at process **703**. Procedure **703** for the addition of passwords is illustrated in *Figure 11*. The password addition process **1101** provides, in combination with the system hardware shown in *Figure 2*, a means for combining the internal three-dimensional data **805** with internal access passwords **1102** defined in the table shown in *Figure 10*. The password addition process **1101** performs a data concatenation to produce an internal file with passwords **1103**.

[0066] After the internal file with passwords has been created, a file key is received at step **704**. This file key is specific to the new file, thereby ensuring that a key providing access to one file would not automatically provide access to other similar files. The key may be specified manually by a user and in this example is made up of four bytes. Alternatively, the key may be generated by the program itself or read from a file containing a plurality of possible keys.

Figure 12

[0067] Having received the file key at step **704**, the file key is combined with the program key at step **705**. The program key is known internally to the program itself and it is envisaged that new versions of the program would have different internal program keys. Referring to *Figure 12*, a combine key process **1201** provides, in combination with the system hardware shown in *Figure 2*, a means for combining a file key **1202** with the program key **1203** to produce an encryption key **1204**.

[0068] Process **706** for encrypting the internal file is illustrated in *Figure 13*. As is well known in the art, an encrypting process takes plain text (machine-readable binary data in this example) and processes this plain text, with reference to a key in order to produce cipher text. The cipher text can then be converted back to the plain text without loss either by using the same key that was used for encryption or by using a different key, depending upon the type of encryption being employed.

[0069] The particular type of encryption being performed is not relevant to the present invention, provided that the encryption provides a suitable degree of protection. Presently, the art suggests that a private key consisting of two hundred and fifty-six bits could not be broken in a realistic time scale through brute force alone. However, it is appreciated that advances continue to be made and key size would need to be reviewed when new versions of the program software are released.

[0070] A suitable form of encryption for the present purpose is considered to be that made available under the designation "Twofish".

[0071] Further details concerning Twofish cipher may be found at http://www.counterpane.com/twofish.html. Twofish is a one hundred and twenty eight bit block cipher that accepts a variable length key up to two hundred and fifty six bits. The cipher is a sixteen round Feistel network with a bijective F function made up of four key dependant eight by eight S boxes, are fixed four by four maximum distance separable matrix, a Pseudo-Hademarde transform, bytwise rotations and a carefully designed key schedule.

[0072] The particular type of encryption used is not an essential feature of the present invention. However, in order to be suitable, the data encryption process must provide a sufficient degree of protection, such as that provided by Twofish, in a time frame considered to be acceptable in commercial applications.

Figure 13

[0073] As illustrated in *Figure 13*, the internal file with passwords **1103** is provided to the data encryption process **1301**, the data encryption process **1301** also receives the encryption key **1204** and performs an encryption process to produce an encrypted internal file **1302**.

[0074] In this example, encrypted data is illustrated as being shaded. It is not possible to perform any manipulations with respect to the encrypted

data until this data is decrypted by means of a decryption process performing the opposite process to that performed by the data encryption process **1301**. The encrypted file written back to disc is a binary file.

Figure 14

[0075] Process **707** adds the file key to the header and process **708** then adds the header to the file. These steps are performed by an add header process **1401**, which, in combination with a system hardware, provides means for effecting the combination of the file key to the header and then the addition of the header to the encrypted internal file.

[0076] It is only the file key **1202** that is included in the header, given that the program key **1203** will be known to the program executed by the receiving equipment.

[0077] Consequently, the encrypted internal file **1302** has a header added thereto and said header includes the file key **1202**.

[0078] A resulting coded export file is identified at **1401**. This includes the encrypted data **1402** and the header **1403**. The header **1403** includes a conventional header **1404**, the file key **1202** and a plurality of random entries **1405**. The purpose of the random entries **1405** is to further camouflage the presence of the file key thereby by making its extraction more difficult except for legitimate holders of the program.

[0079] The conventional header **1404** includes an identification showing that the file is binary file and an identification of the file version. In addition, the header includes identification of the version of the program used for its creation and a flag that may be set or unset but when set confirms that the remainder of the file has been encrypted.

Figure 15

[0080] The coded export file **1401** is now available for distribution, as shown in *Figure 15*. The export file is distributed to consumers **1501**, prosumers **1502** and professionals **1503**. In theory, anyone can receive encrypted files from any source but in most cases consumers would receive files over the Internet or by data carrying media such as CD ROMS and DVDs. Similarly, prosumers receive files from data distributors, such as data distributor **1504** via the Internet or via CD ROMS and DVDs etc.

[0081] Professionals, who may be in-house professionals, receive data via internal networks, via the Internet, using re-writable media such as zip drives and by using read-only media such as DVDs.

[0082] Different levels of access are usually given to different types of recipients. Thus, in this example, consumers **1501** are given access to Levels A, B and C. However, it is expected that consumers having access to Level B will have to make some payment and to be given access to Level C a higher payment would be required.

[0083] A further payment would be made by prosumers, thereby giving them access to Level D. Professionals assisting in the development of a project would require access to Levels E, F or G as considered appropriate.

[0084] After receiving a file from a data distributor or a colleague, it is necessary to perform a decoding operation **413** in order to effect further data manipulation. In this example, animation studio **102** has generated a protected file that has been supplied to effects studio **103**. Further manipulations have been performed by the effects studio **103** and a protected file has then been returned back to the animation studio **102** and in particular to the station illustrated in *Figure 2*. The station in *Figure 2* will now be described with respect to the performance of the decoding operation **413**, as illustrated in *Figure 16*.

Figure 16

[0085] At step **1601** the program is opened and at step **1602** the encoded file is received. At step **1603** the receiving user enters the password which would either be WORD B, WORD C, WORD D, WORD E, WORD F or WORD G depending on the level of access afforded them. In this example, the file has been returned back to the animation studio **102** so the user would enter WORD G. Had the file been received at the customer centre **105** no password would be entered but only Level A access would be obtained. Had the file been sent to the texture studio **104**, the user would enter WORD F and would not be capable of gaining the full control of Level G.

[0086] At step **1604**, the file is decoded and at step **1605** the level of access is acknowledged to the user.

[0087] At step **1606** a question is asked as to whether the data is to be rendered and displayed which would usually be answered in the affirmative, particularly given that a password is not required in order to achieve this. Consequently, the data is rendered and displayed at step **1607**.

[0088] At step **1608** a question is asked as to whether the data is to be manipulated and this question may be answered in the negative if the user does not have a level of access above Level A. Assuming a user does have access above Level A, the question asked at step **1608** is answered in the affirmative allowing data manipulation to be performed at step **1609**.

[0089] At step **1610** a question is asked as to whether a file is to be exported, ie requiring the encoding operation **412** and if answered in the affirmative, a file is exported at step **1611**. If a user does not have access above Level A, it is not possible for the question asked at step **1610** to be answered in the affirmative thereby such users cannot export files.

[0090] If any of questions **1606**, **1608** and **1610** are answered in the negative, control is directed to step **1612** where a question is asked as to whether another file is present. If answered in the affirmative, control is returned to step **1602** alternatively control is directed to step **1613** and the program is closed.

Figure 17

[0091] Step **1603** invites a user to enter a password, resulting in an invitation being displayed on monitor **207**, as illustrated in *Figure 17*. A user is invited to enter a password in text box **1701** and then effect a pressing of soft button **1702** to continue or button **1703** to cancel.

Figure 18

[0092] Process **1604** for the decoding of an encoded file is detailed in *Figure 18*. At step **1801** the file key is extracted from the header, given that the decoding operation **413** is aware of the location of the file key bytes within the header.

[0093] At step **1802** the file key is combined with the program key; again the decoding operation **413** being aware of the program key.

[0094] At step **1803** the data is decrypted by means of a process performing the reverse operation to encryption process **1301**.

[0095] At step **1805** the level of access available to a user is determined by comparing the access password entered at step **1603** with passwords stored within the decrypted file.

[0096] At step **1806** the decrypted data is internalised and access level flags are set. Thus, process **1806** represents the data being transferred from the decoding operation **413** to the data manipulation

region that includes manipulation programs **401** to **405** etc. The individual data elements are also considered independently as such allowing appropriate programs to be operated upon each one of them independently.

Figure 19

[0097] The overall process for receiving an encoded file and then acting upon it is illustrated in *Figure 19*. Extraction process **1801** provides, in combination with the system hardware, means for receiving coded import file **1401** and producing an encrypted file **1802**, similar to encrypted internal file **1302**.

[0098] Decryption process **1803**, in combination with the system hardware, provides means for decrypting encrypted file **1802** to produce a decrypted file **1804**, being an internal file with passwords similar to file **1103**.

[0099] Access process **1805** provides, in combination with the system hardware, means for receiving a decrypted file **1804** and producing a combined image data file **1806**, similar to file **805** along with access codes **1807**, similar to codes **1102**.

[0100] A combination of processes **801** to extract, **1803** to decrypt and **1805** to access may be considered in combination as the decoding operation.

[0101] Process **1807** splits the combined data 1806 into its individual elements **1808**, similar to elements **801**, **802** and **803**. Once placed in this form, it is then possible for manipulations to be performed, dependent upon the level of access provided by the access codes **1807** and the user's ability to enter an access password.

Figure 20

[0102] Having performed the decoding operation, the animation studio **102** is then in a position to review the work done by studio **103** and make further manipulations. Consequently, the mixed-media data is rendered and displayed on monitor **207**, as illustrated in *Figure 20*. Thus, as shown in *Figure 20*, further manipulations have been made to the original data, as displayed in *Figure 6* and, with a requisite level of access, it is now possible for the animation to perform further manipulations and possibly complete the project. Thereafter, three-dimensional data may be released and distributed as illustrated in *Figure 15* but with different types of user being allocated appropriate access codes thereby ensuring that anyone who has not been given permission to access appropriate levels of the data are prevented from doing so.